Amendments to the Specification:

Please amend paragraphs [0014], [0015], and [0016] of the present specification as follows:

[0014] Because the statistical record purge procedure takes a finite amount of time and results in the statistics database being inaccessible to the statistics server(s) while the procedure is running, the temporary memory or log space for the statistics server(s) must have sufficient room to store all the new records generated by the statistics server(s) while the purge procedure is running. Otherwise, potentially important newly generated statistical records may be lost due to log space overflow. In an exemplary embodiment of the invention the additional instructions calculate whether there is 90% or more log space free before calling on the purge script to execute. If not, the scheduler on the fault server waits for approximately 30 seconds before checking the free log space again. The amount of free log space that is required will necessarily depend on a number of factors such as the number of statistical records likely to be generated during the time the purge script is running, how often the purge procedure is run (e.g., hourly, bi-hourly, daily, etc.), the size of the log space (e.g., 2052 Megabytes in the exemplary embodiment noted above), the size of the statistical records generated by the statistics server(s), and the like. Pseudocode for these instructions in accordance with an exemplary embodiment of the invention appear in ~~Appendix~~ Source Code Listing A ~~attached hereto~~ shown below. Also, Fig. 3 illustrates in flow diagram form the method outlined above.

[0015] Note also that in the exemplary embodiment illustrated in ~~Appendix~~ Source Code Listing A, statistical records are maintained in the statistics database(s) for 30 days, the purge procedure is run once daily, and it takes approximately 20 minutes for the purge script to execute once it has been called by the scheduler on the fault server(s).

[0016] Another improvement to existing statistical record purge procedures that may be implemented is to update the pointers and/or indices in the statistics database(s) following the execution of the purge script. Updating the pointers results in notably faster read and write operations in the statistics database(s). Pseudocode for the instructions to update the pointers and indices in accordance with an exemplary embodiment of the invention are ~~attached hereto~~ shown below as ~~Appendix~~ Source Code Listing B.

Please add the following tables (i.e. Source Code Listing A and Source Code Listing B) to the end of paragraph [0016]:

Source Code Listing A

check_purge.sh

```
prog=`basename $0`

if test $# -lt 1
then
        echo "Need dbname" >>/opt/BulkStats/etc/$prog.log
        exit 1
fi

if test -s /opt/BulkStats/etc/$prog.log
then
        dte=`date +%d%b%Y`
        mv -f /opt/BulkStats/etc/$prog.log \
                /opt2/BulkStats.var/$prog.log@$dte
        compress -f /opt2/BulkStats.var/$prog.log@$dte
fi

DBNAME=NAVIS-STATN
export prog DBNAME

if ping -I 1 navis-statn 24 1|grep "0 packets received"
then
     echo "navis-statn not responding at `date`" \
          >>/opt/BulkStats/etc/$prog.log 2>&1
     rm -f /BulkStats/data/NXStatisticsCbxGbx.purging
     exit 1
fi
```

```
######################################
# lock out other db type cron jobs !
######################################
touch /BulkStats/data/NXStatisticsCbxGbx.purging

###########################################################
# this is a routine to check for an empty db log, if not
# sleep up to 10 minutes waiting for one
###########################################################
check_db ()
{
        ##############################################################

        # loop up to 12 times, i.e. 6 minutes, until the logfile is

        # close to 100% free
        ##############################################################


        cnt=12
        while true
        do
remsh $DBNAME -l sybase -e /opt/sybase/query >/tmp/$prog.$$ 2>&1 <<!

sp_helpdb $1
go
quit
exit
!
LogSize=`cat /tmp/$prog.$$ | grep _log | awk ‘{print $2,$6}’`
                rm -f /tmp/$prog.$$

                Size=`echo $LogSize | awk ‘{print $1}’ | cut -f1 -d’.’`
                Free=`echo $LogSize | awk ‘{print $2}’`

                Size=`expr $Size \* 1024000`
                Free=`expr $Free \* 100000`

                WFree=`expr $Free \/ $Size`
                RFree=`expr $Free \% $Size`
                RFree=`echo $RFree | cut -c1-2`

                echo “$1 has $WFree.$RFree free logspace at `date`” \
                        >>/opt/BulkStats/etc/$prog.log 2>&1

                if test $WFree -gt 85
                then
                        echo >>/opt/BulkStats/etc/$prog.log 2>&1
                        break
                else
                        sleep 30
                fi

                cnt=`expr $cnt - 1`
                if test $cnt -lt 0
                then
                        echo “$prog: aborting because of full db log for $1” \
                                >>/opt/BulkStats/etc/$prog.log 2>&1
                fi
        done
}

export MinTime MaxTime
echo “$prog:\tStarting at `date`\n” >>/opt/BulkStats/etc/$prog.log
check_db “$1”

for x in TrkStat CktStat TrunkStat FrCktStat FrLportStat ATMCktStat ATMPrtStat \
        ATMSvcStat ATMTrkStat ATMLPrtNiStat ATMLPrtTrkStat ATMFirstTrkStat \
        ATMOptTrkStat IpLportStat SmdsLportStat
do

        MinTime=`remsh $DBNAME -l sybase -e /opt/sybase/query <<!
use $1
go
select min(startTime) from $x
go
quit
exit
!`
        if echo $MinTime | egrep “NULL|Msg” >/dev/null
        then
echo “No table data for $x\n” >>/opt/BulkStats/etc/$prog.log
                continue
        fi

        MinTime=`echo $MinTime | awk ‘{print $2}’`
        echo “$x:\tMinTime <$MinTime>“ >>/opt/BulkStats/etc/$prog.log
        if test $MinTime -le 950000000
        then
                echo “$x:\tbad number for MinTime” \
                        >>/opt/BulkStats/etc/$prog.log
                continue
        fi

        MaxTime=`/BulkStats/bin/perl5 -e ‘$utcseconds=time();print “$utcseconds\n”’`
        echo “$x:\tMaxTime <$MaxTime>“ >>/opt/BulkStats/etc/$prog.log
        if test $MaxTime -le 950000000
        then
```

```
                echo “$x:\tbad number for MaxTime” \
                        >>/opt/BulkStats/etc/$prog.log
                continue
        fi

        DiffTime=`expr $MaxTime - $MinTime`
        DiffTime=`expr $DiffTime / 86400`

        echo “$x:\tnumber of days in database is $DiffTime\n” \
                        >>/opt/BulkStats/etc/$prog.log

        ###########################################
        # delete all records older than 30 days
        ###########################################
        if test $DiffTime -gt 31
        then
                Ttime=`expr $DiffTime - 31`
                DelTime=0
                export DelTime

                while true
                do
                        if test $Ttime -eq 0
                        then
                                break
                        fi

                        DelTime=`expr “$MinTime” + 86400`
                        MinTime=`expr “$MinTime” + 86400`
                        export DelTime MinTime

                        echo “$x:\tDelTime <$DelTime>“ \
                                >>/opt/BulkStats/etc/$prog.log
                        echo “$x:\tdelete $x where startTime < $DelTime at `date`\n” \
                                >>/opt/BulkStats/etc/$prog.log

                        ###################################################
                        # execute the ‘query’ file on remote server so
                        # passwd is not exposed !
                        ###################################################
                        remsh $DBNAME -l sybase -e /opt/sybase/query \
                                >>/opt/BulkStats/etc/$prog.log 2>&1 <<!
use $1
go
delete $x where startTime < $DelTime
go
checkpoint
go
quit
exit
```

```
!
            DiffTime=`expr $DiffTime - 1`
            echo "\n$x:\tnumber of days left in database is $DiffTime" \
                >>/opt/BulkStats/etc/$prog.log

            Ttime=`expr $Ttime - 1`

            echo >>/opt/BulkStats/etc/$prog.log
            check_db "$1"
        done
    fi
done

rm -f /BulkStats/data/NXStatisticsCbxGbx.purging
echo "$prog:\tEnding at `date`\n" >>/opt/BulkStats/etc/$prog.log
```

Source Code Listing B

check_ stats.sh

```
prog=`basename $0`

if test $# -lt 1
then
    echo "Need dbname" >>/opt/BulkStats/etc/$prog.log
    exit 1
fi

>/opt/BulkStats/etc/$prog.log

##if test -s /opt/BulkStats/etc/$prog.log
##then
    ##mv -f /opt/BulkStats/etc/$prog.log \
        ##/opt/BulkStats/etc/$prog.log.old
##fi

DBNAME=NAVIS-STATN
export prog DBNAME
echo >>/opt/BulkStats/etc/$prog.log

for x in TrkStat CktStat TrunkStat FrCktStat FrLportStat ATMCktStat ATMPrtStat \
    ATMSvcStat ATMTrkStat ATMLPrtNiStat ATMLPrtTrkStat ATMFirstTrkStat \
    ATMOptTrkStat IpLportStat SmdsLportStat
do

    echo "Starting update statistics $x at `date`" >>/opt/BulkStats/etc/$prog.log
    remsh $DBNAME -l sybase -e /opt/sybase/query \
                >>/opt/BulkStats/etc/$prog.log 2>&1 <<!
use $1
go
update statistics $x
go
quit
exit
!
        echo "Ending update statistics $x at `date`\n" \
                >>/opt/BulkStats/etc/$prog.log

done

echo "$prog:\tEnding at `date`\n" >>/opt/BulkStats/etc/$prog.log
```

Please amend the Abstract of the disclosure as follows:

Improvements to existing statistical record memory purge procedures and processes are shown and described. Certain procedures ensure that there is adequate temporary memory for storing new statistical records before purging older statistical records from permanent memory. Certain of the procedures wait a predetermined amount of time before reevaluating whether there is adequate temporary memory for storing new statistical records. After older statistical records have been purged, certain of the procedures update any indices or pointers utilized by permanent memory. The improvements may be implemented in a telecommunications system having a plurality of managed elements, each of the managed elements potentially generating statistical data that is communicated to one or more statistics servers.